WHAT IS CLAIMED IS:

1. A method for obtaining an average of a plurality of values, wherein a first plurality of values is stored in a first packed structure, wherein a second plurality of values is stored in a second packed structure, the method comprising

using an averaging operation on the values in the first and second packed structures to obtain a plurality of values in a packed average result, wherein a value in the packed average result equals a rounded-up average of a value in the first packed structure and a value in the second packed structure;

determining whether the sum of the value in the first packed structure plus the value in the second packed structure is an odd number and, if so, performing the step of

subtracting one from the value in the packed average result to obtain a packed adjusted result.

2. The method of claim 1, wherein the step of using an averaging operation includes

using a single-instruction multiple data operation.

3. The method of claim 2, wherein the step of using a single-instruction multiple-data operation includes

using a *PAVG* instruction.

4. The method of claim 1, wherein the step of determining whether the sum of the value in the first packed structure plus the value in the second packed structure is an odd number comprises substeps of

performing an exclusive-or operation on values in the first and second packed structures to obtain a packed exclusive-or result;

masking all but the least significant bit (lsb) of values in the packed exclusive-or result to obtain a packed lsb result; and using the packed lsb result in the step of subtracting one from the value in the packed average result.

5. The method of claim 1, wherein a structure includes a word of memory.

6. The method of claim 1, wherein a structure includes a register.

7. The method of claim 1, wherein the packed adjusted result is used in the derivation of a finite impulse response filter.

8. The method of claim 1, wherein each structure comprises 8 values of 8 bits each.

9. A method for adjusting the result of a *PAVG* instruction, wherein a first set, A, of packed values, a_i, and a second set, B, of packed values, b_i, are operated on by *PAVG* to obtain $PAVG(A,B) = [(a_i+b_i+1) >> 1, i=1,\ldots,8]$, the method comprising

adjusting the result of the *PAVG* operation to obtain a packed value result, C, as $C = PAVG(A, B) - (A \wedge B)$ & 0x01.

10. A method for achieving an averaged result on packed binary values A_1,A_2, A_3,A_4, the method using a *PAVG* instruction that computes a rounded-up average on first and second sets of packed values to produce a resulting set of packed averages, wherein $B_1 = PAVG(A_1,A_2)$ and $B_2 = PAVG(A_3,A_4)$, the method comprising

deriving a result, R, as

$$R = \begin{cases} PAVG(B_1,B_2) - (B_1 \wedge B_2) \,\&\, ONE & \text{when } E = 0 \\ PAVG(B_1,B_2) & \text{when } E = 1 \end{cases}$$

wherein ONE is a value with a one in the least significant bit position of one or more packed values and wherein $E = 1$ when both (A_1+A_2+ONE) and (A_3+A_4+ONE) are odd integers.

11. The method of claim 10, wherein $EB_1 = (A_1 \wedge A_2)$ and $EB_2 = (A_3 \wedge A_4)$, and wherein

$E = \sim(A_1 \wedge A_2)$ & $\sim(A_3 \wedge A_4)$ & $ONE = \sim(EB_1 \mid EB_2)$ & ONE.

12. The method of claim 10, wherein the step of deriving includes deriving R as $R = PAVG(B_1,B_2) - (B_1 \wedge B_2)$ & $\sim E$ & ONE.

13. The method of claim 10, wherein the step of deriving includes deriving R as $R = PAVG(B_1,B_2) - (B_1 \wedge B_2)$ & $((A_1 \wedge A_2) \mid (A_3 \wedge A_4))$ & ONE.

14. A method for achieving an approximate averaged result on packed binary values A_1,A_2, A_3,A_4, for use in finite impulse response filter computations, the method using a *PAVG* instruction that computes a rounded-up average on first and second sets of packed values to produce a resulting set of packed averages, wherein $B_1 = PAVG(A_1,A_2)$ and $B_2 = PAVG(A_3,A_4)$, the method comprising

deriving a result, R, as

$$R = PAVG(B_1,B_2) - (B_1 \wedge B_2) \text{ \& } ONE.$$

wherein ONE is a value with a one in the least significant bit position of one or more packed values.

15. A method for achieving an averaged result on packed binary values for use in finite impulse response filter computations, the method using an instruction, *PAVG*, that computes a rounded-up average on first and second sets of packed values to produce a resulting set of packed averages, the method comprising

detecting when the use of the *PAVG* instruction introduces a rounding-up increase in an averaged result; and

decreasing the rounding-up increase to achieve a desired result.

16. The method of claim 15, wherein the step of detecting further comprises detecting when one or more averaged results are odd.

17. The method of claim 15, wherein an exact desired result is achieved.

18. The method of claim 15, wherein an approximate desired result is achieved.

19. The method of claim 18, further comprising

assuming that a significant bit of at least one packed value averaged result of the *PAVG* instruction is a 0.

20. The method of claim 18, further comprising

assuming that a significant bit of at least one packed value averaged result of the *PAVG* instruction is a 0.

21. A method for using a single-instruction multiple-data (SIMD) instruction to perform a function, wherein the SIMD instruction uses M arguments, wherein the function uses N variables, wherein M and N are not the same, the method comprising

using the SIMD instruction on a plurality of packed values to obtain an inaccurate packed value result; and

adjusting the inaccurate packed value result to obtain an adjusted packed value result.

22. The method of claim 21, wherein the adjusted packed value result is an approximate result.

23. The method of claim 22, further comprising

correcting the adjusted packed value result to obtain a desired packed value result.

24. The method of claim 21, wherein the SIMD instruction includes an averaging operation

25. The method of claim 22, wherein the step of using the SIMD instruction includes

using a *PAVG* instruction.